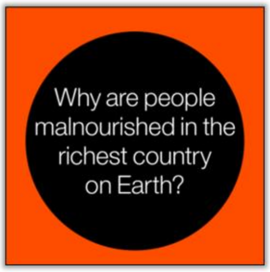
Hello. We are project team 25. Today we are going to share with you the results and recommendations from our project called the "Apple-o-matic Service"

System Problem Statement

- To satisfy a child's hunger quickly and nutritiously



MITsdm system design and management

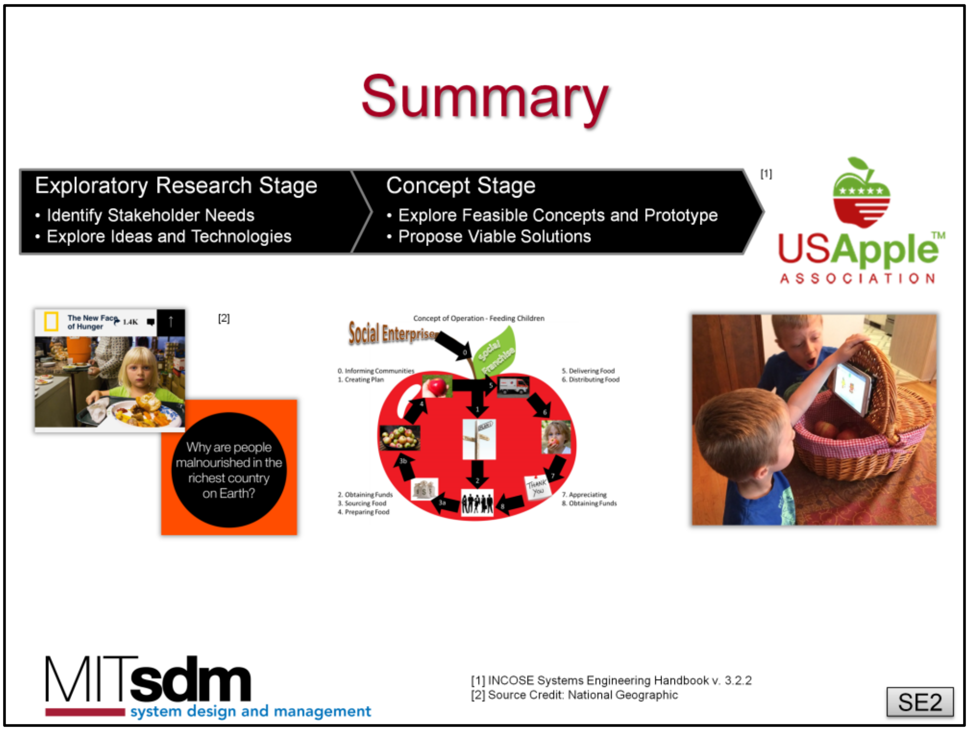
SA 3

Katrina -

15.8 million children in the U.S. living a food-insecure households (2012 data) is a big problem due to sociotechnical challenges when implementing a solution.

The challenge for our team was to take this SPS and generate our Goal Statement from this problem

SA3: A system problem statement and an explicit system boundary.



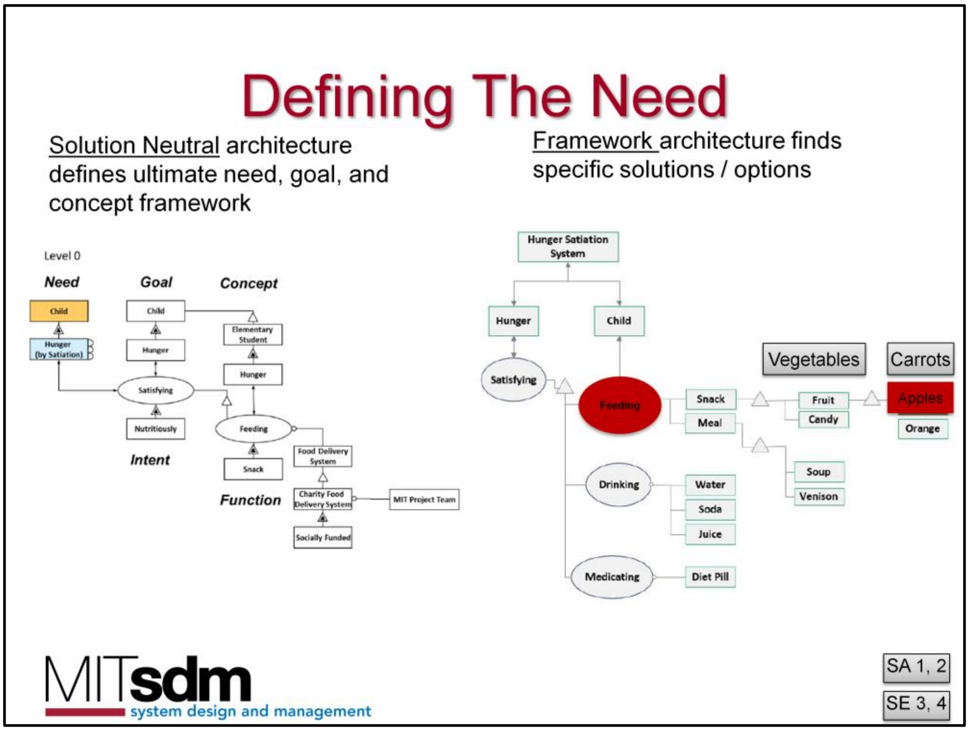
Katrina Speaking –
We designed a prototype solution that we demonstrated in a feasibility evaluation. It was our goal to package the deliverable into a plan for handoff to an organization such as USApple association.

During our Spring Term Project, we successfully practiced using many of the SDM learning objectives to perform initial product development lifecycle stages to help solve this hunger issue . This presentation will explain how we

1) Identified Stakeholder Needs
2) Explored Ideas and Technologies
3) Generated a feasible Concept of Operation with a prototype and
4) Finally the recommendations for a viable solutions base on the result of our 1 week prototype evaluation with 22 first graders in Dubuque, Iowa.

Our team was motivated by the problem scope to use Systems Design & Management methods and tools. 15.8 children living a food-insecure households is a big problem due to sociotechnical challenges when implementing a solution like the Apple-o-matic Service. We needed a set of tools to systematically walk through down a design path to quickly produce a feasible prototype and project plan. Most importantly we wanted to power change that makes a positive impact in our communities.

SE2: Systems Engineering methods and tools used – why did you choose them and comments on methods you considered not appropriate for your project.



Katrina -

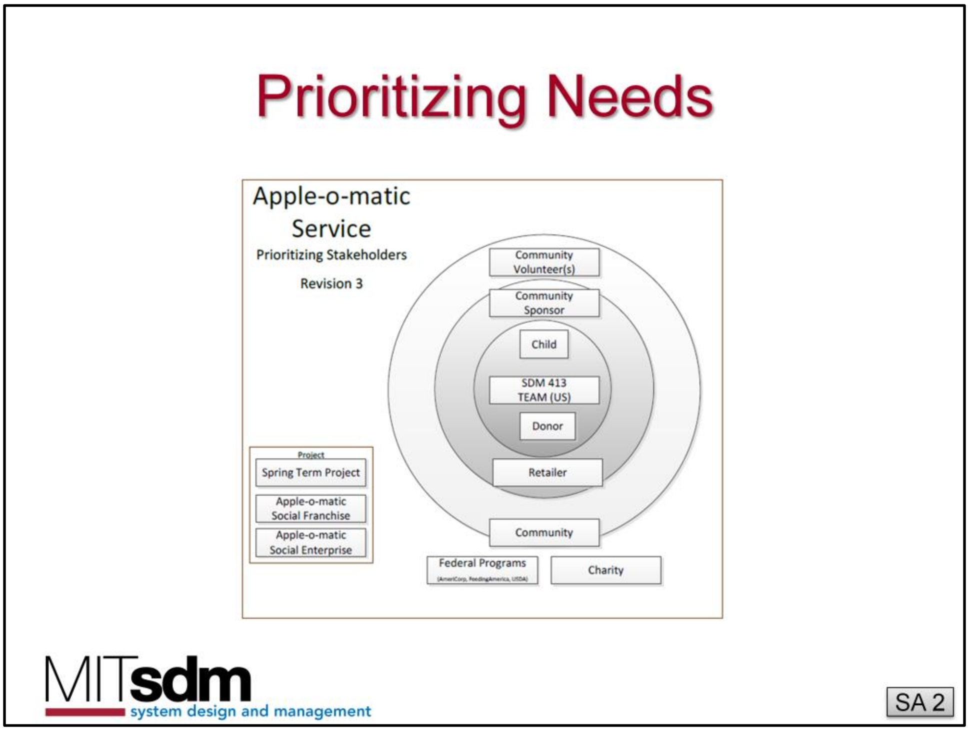
In order to start this project, and to fulfill the high level need and our motivation to succeed, we defined the true need. We broke it down and found the solution neutral need of the children. We architected a high level goal and concept.

We then also used that information to look at other concepts that could meet or suffice the “solution neutral need”. While some of these other concepts were absurd, we wanted to explore possible other options to an apple as a hunger satiating form.

From the high level architectural decisions, we chose to use apples as the form.

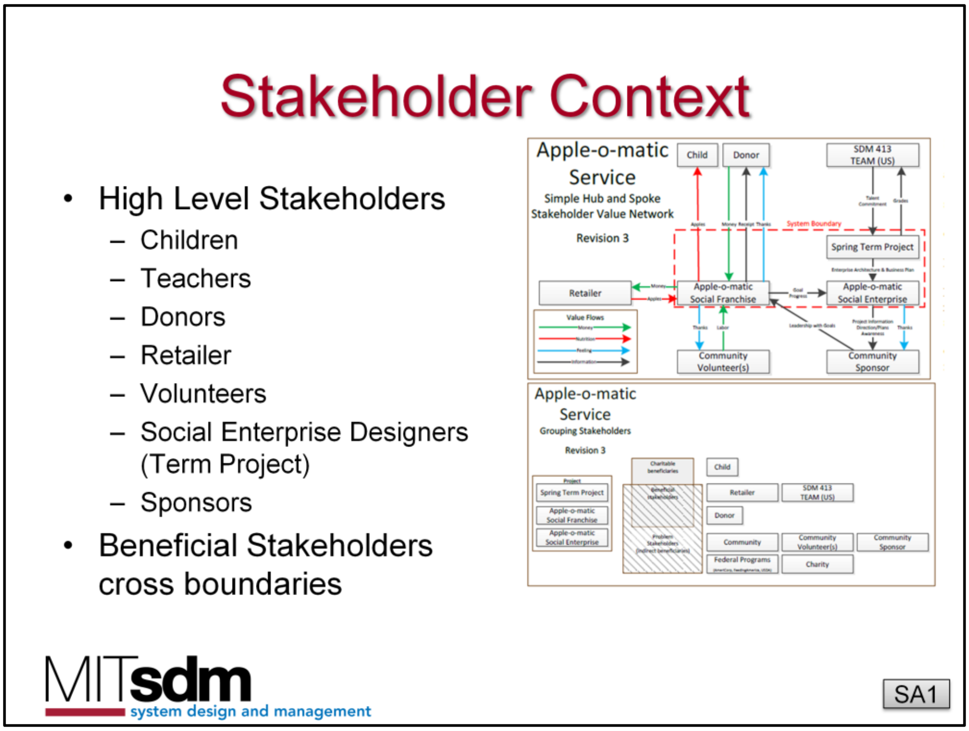
As we rolled out the feedback from our feasibility study, we introduced another concept of using vegetables based on suggestion from the teacher .

SA1: A representation of the system architecture, including the architectural decisions that the team made and why.
SA2: A representation of the stakeholders and their needs, and an accompanying analysis of value delivery and prioritization of stakeholder and/or goals.
SE3: Different concepts you generated.
SE4: Concept selection process.



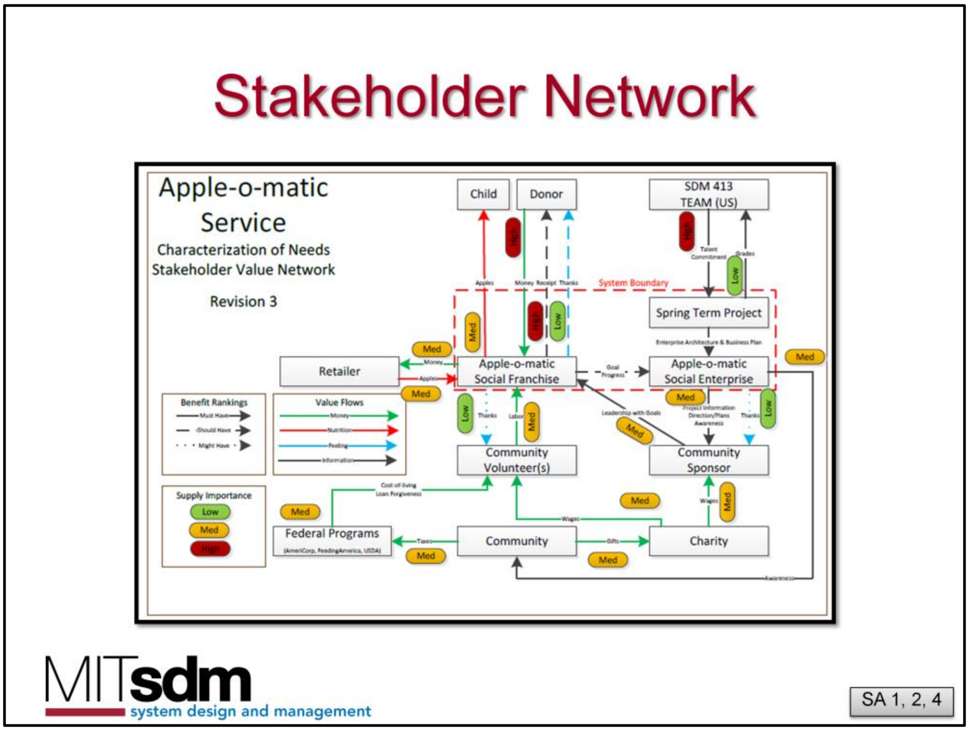
Mapping the prioritizations helped our group apply the right resources, as well as ensure that the value related functions within Apple-o-matic achieve and meet the intent of our goals.

SA2: A representation of the stakeholders and their needs, and an accompanying analysis of value delivery and prioritization of stakeholder and/or goals.



High Level stakeholders as shown in the prioritization are shown here. We were able to identify where our system boundary interacts with the stakeholders on the outside.

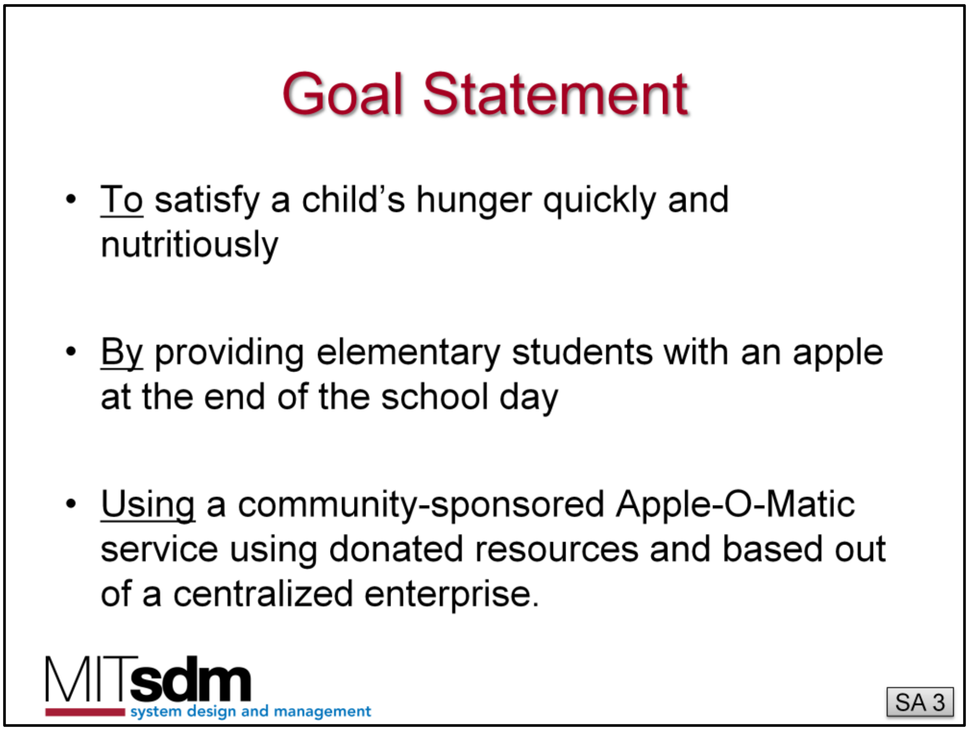
SA1: A representation of the system architecture, including the architectural decisions that the team made and why.



Katrina -

And this stakeholder value network graph shows prioritization models on the system architecture.   Highlights of this system include: Donor to AoM, Child to AoM and subsequently the Community Sponsor and AoM plus the Retailer. These are all higher ranking stakeholders in our system.

SA1: A representation of the system architecture, including the architectural decisions that the team made and why.
SA2: A representation of the stakeholders and their needs, and an accompanying analysis of value delivery and prioritization of stakeholder and/or goals.
SA4: A description of the broader contact in which the system sits, and the whole product context (including legal and standards).

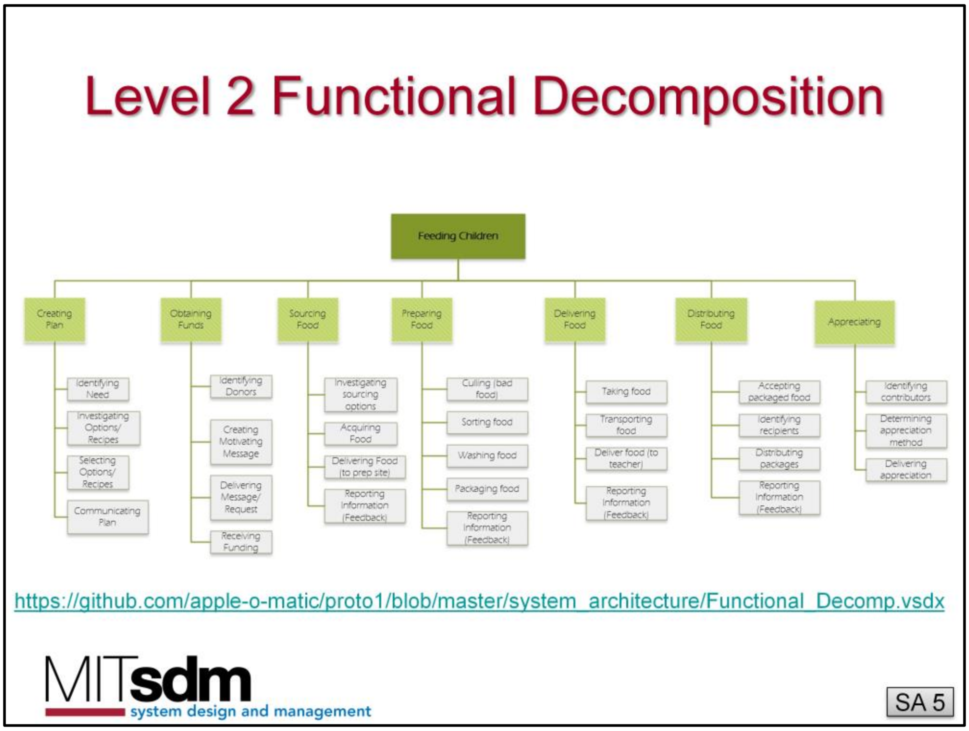


Katrina -

After analyzing and understanding the solution neutral framework, applying prioritizations, and really understanding the needs of the system stakeholders, we were able to put together the system problem statement.

This is the rest of the challenge…

SA3: A system problem statement and an explicit system boundary.

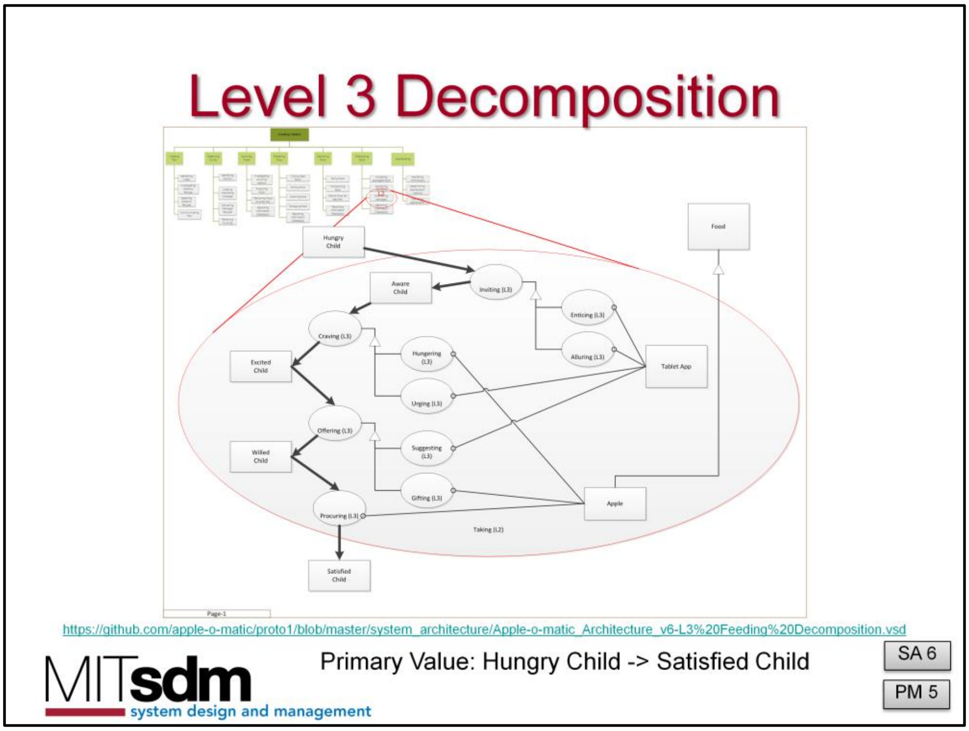


Katrina –

Now lets go through the architecture to show how we got there.

-Using the stakeholder network and prioritized needs, used this decomposition to look at dependencies within the system, which showed us

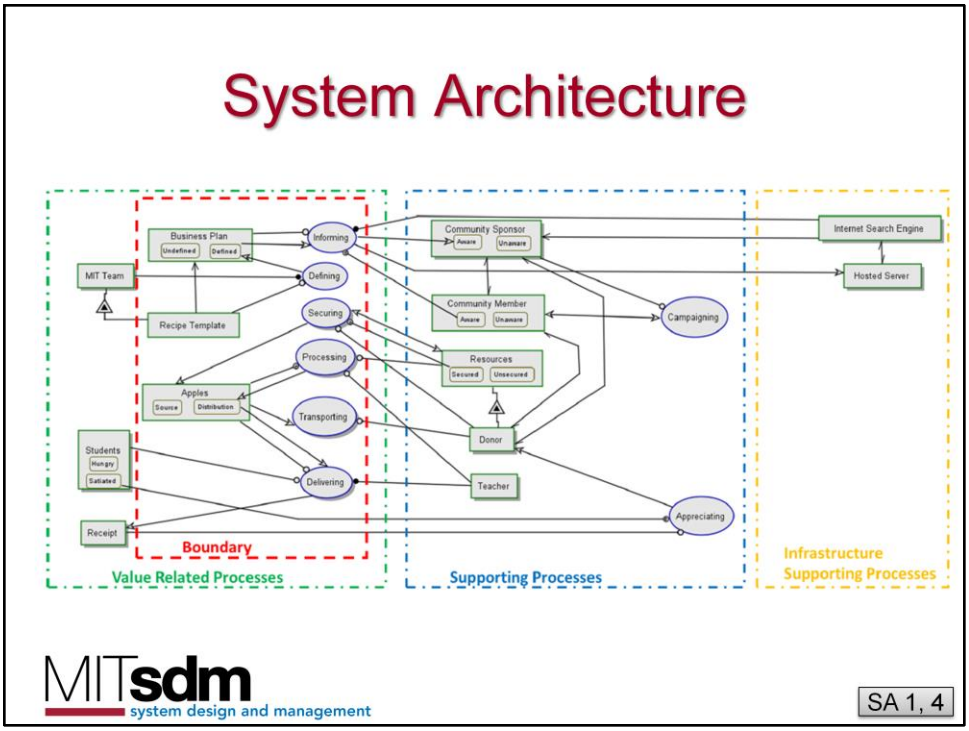
SA5: A functional description of the system, with at least two layers of decomposition



Katrina Speaking –

The Level 3 Decomposition of the critical function provided enough detail to design a Tablet App and requirements used in Feasibility Evaluation. The critical function of “Distributing Packages” was decomposed to identify the **primary value path**.

SA6: The decomposition of form to two levels of detail, the allocation of function to form, and the structure of the form at this level.
PM5: Any design, prototyping, and validation tasks by the project team to date are reflected in an estimate of future implementation activities

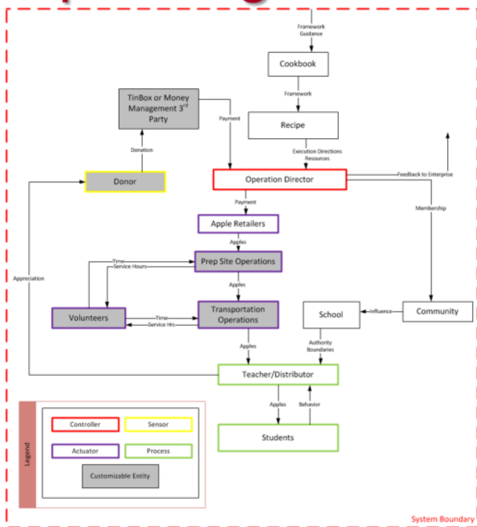


Katrina –

OPCAT was used to define and relate functions and form to one another. Systems that exist and that our service would utilize are in the infrastructure supporting processes such as the internet and GitHub. Our recipe and system definition resides in its own boundary, but is supported by processes that exist in almost any community in the US (grocery stores, farmers, teachers, and schools).

SA1: A representation of the system architecture, including the architectural decisions that the team made and why.
SA4: A description of the broader contact in which the system sits, and the whole product context (including legal and standards).

Operating Process



https://github.com/apple-o-matic/proto1/blob/master/system_engineering/Control%20Structures.vsd

MITsdm system design and management

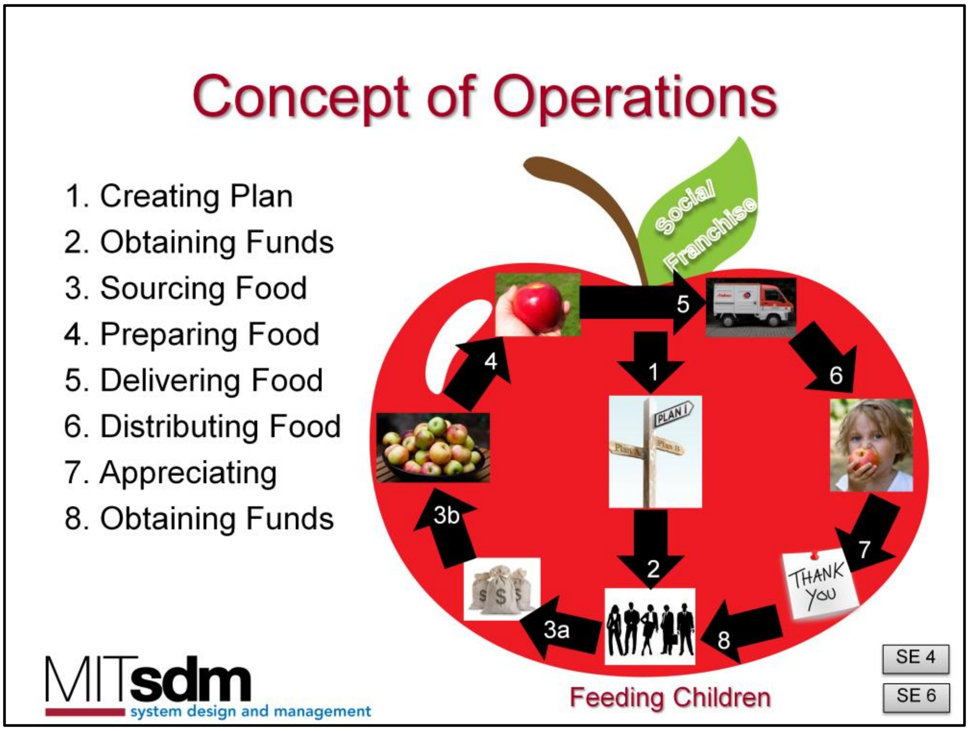
SE 1, 2

PM 4

Aaron - The Operating process control structure illustrates the local "recipe" control, and exhibits the system boundary within which our project was focused.

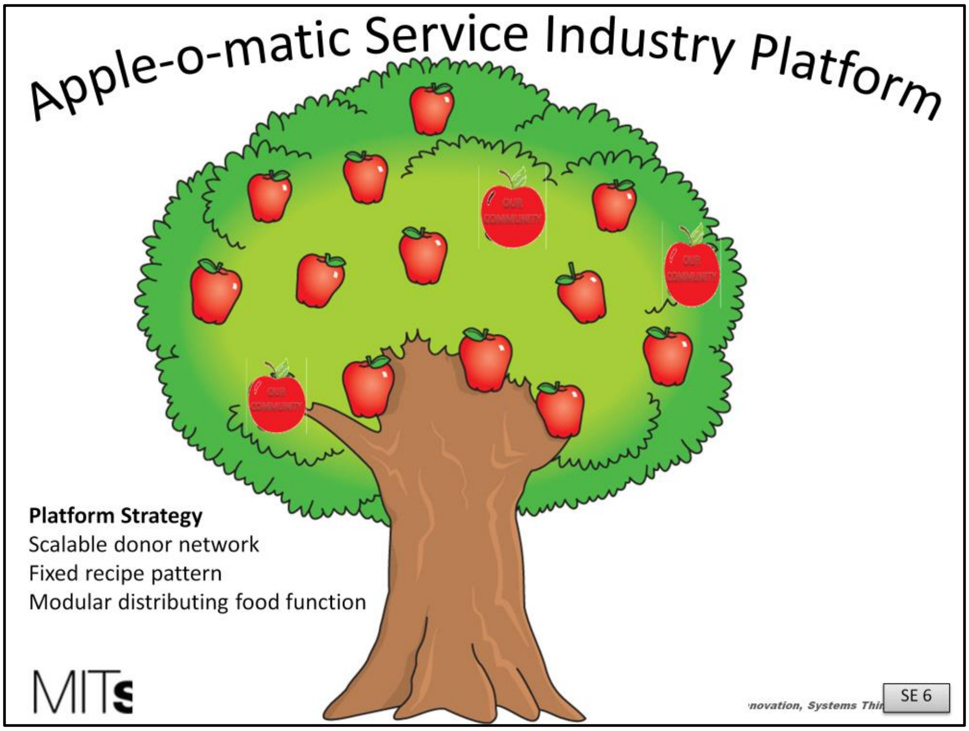
This is the "engineering" view of the operating process, but to make an operating process graphic that is more acceptable….(TRANSITION to next slide)

SE1: System requirements including requirements traceability from stakeholder needs to subsystem and component requirements.
SE2: Systems Engineering methods and tools used – why did you choose them and comments on methods you considered not appropriate for your project.
PM4: Choices for structuring and integrating the product system, workflow process, and project organization are highlighted, particularly those choices which are exceptions to standards or conventional wisdom.



Aaron -
The operations concept Begins with Informing Communities. By way of the internet and charitable organizations (such as USApples), they will reach out to the public. The school or beneficiary advocate will decide on a recipe plan (ref 1.0). Once the system in in place from the recipe, donors will donate funds (2.0). The system will move funds to the source for food (such as a grocer) and purchase the apples (3.0). The apples are prepared by the vendor, and modularized to allow local growers to prepare food for delivery (4.0). Once ready, they are picked up and delivered by a variety of transportation methods (such as volunteer or vendor specific tucks 5.0). They are distributed to the schools, teachers and or volunteers will be required to hand out apples or monitor their method of delivery to students (vending) shown in (6.0). The system is provisioned for feedback to appreciate the original donors by Apple-o-matic and hope that the Donors in 2.0 donate money again.

SE4: Concept selection process.
SE6: Operations and lifecycle considerations.



Aaron –
The Apple-o-matic Social Enterprise is a firm that is analogous to a tree with supportive branches. The Apple-o-matic Social Enterprise creates a new Social Franchise with a strong connection to a community. Mention ***RECIPES***

Our architecture intent and system is intended to be modularized and scalable to a larger network. Analogous to an apple tree, each apple (an apple-o-matic product) can be seen and “grown” anywhere in the continental US.
This platform overlays on existing programs supported at the national level such as farm-to-school and National School Lunch Program.

Architectural Decisions

Architectural Decision Question	Mutually Exclusive Options
Distribution Method?	• **Teacher is primary distributor** • Vending Machine as distributor • Bushel Basket as distributor • Cafeteria at school is primary distributor • After school program is primary distributor • Bus Driver is primary distributor • Parent is primary distributor • Postal service is primary distributor
What to distribute?	• **Snack** • Meal • Drink • Medication
Enterprise Business Type	• **Nonprofit Organization** • Charity • Lucrative business model • Illegal business (cartel)
How to coordinate/facilitate resources?	• **Central Phone application** • Flyers • Crowd sourced funding (DonorsChoose.org)
Brand governance	• **Social franchise, centrally controlled** • Autonomous, anarchy
Who to distribute to?	• **Hungry children** • Malnourished children • Food insecure children • All children • School children • Adults

MITsdm
system design and management

SA 1

SA1: A representation of the system architecture, including the architectural decisions that the team made and why.

Key Requirements

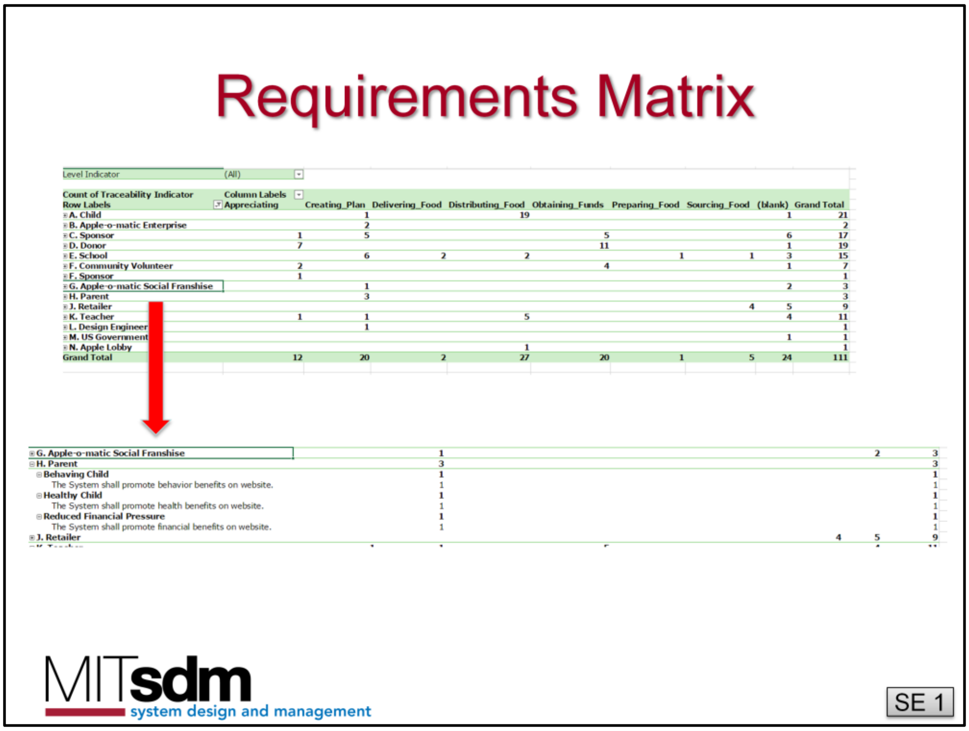
ID	Stakeholders	Needs	System Requirements
A.A.1	A. Child	Hunger Satisfied	The System shall feed apples during time of hunger.
A.A.2	A. Child	Nutritious snack	The system shall make healthy snacks available to children.
A.B.1	A. Child	Balanced Nutrition	The system shall provide a nutritious snack
B.A.1	B. Apple-o-matic Enterprise	Enterprise Architecture	The system shall have a clearly defined architecture.
B.B.1	B. Apple-o-matic Enterprise	Business Plan	The system shall have a clearly defined business plan.
C.B.1	C. Sponsor	Confidence of Actual Impact	The System shall publish metrics for existing community implementation.
C.D.1	C. Sponsor	Coordination Skills/Tool	The system shall provide templates to setup social franchise
C.D.2	C. Sponsor	Easy setup of project	The system shall be easy to setup.
C.E.1	C. Sponsor	Personally Inexpensive project	The System shall not require monetary distribution from sponsor.
D.A.1	D. Donor	Tax Deduction	The System shall be a tax exempt entity.
D.B.1	D. Donor	compelling message to donate	System administrator shall create compelling message to donate, to be delivered to potential audience.
D.C.1	D. Donor	Feedback from beneficiary	The donor shall have the ability to receive feedback from teacher and or students
D.D.1	D. Donor	Safe money handling	The system shall manage donor funds appropriately
E.A.1	E. School	Safe food for children	The system shall ensure security of the food provided to children.

https://github.com/apple-o-matic/proto1/blob/master/system_engineering/Apple-o-matic_Requirements.xlsx

MITsdm system design and management

SE 1

Aaron – This are key requirements to illustrate the mapping from stakeholder needs to system requirements.



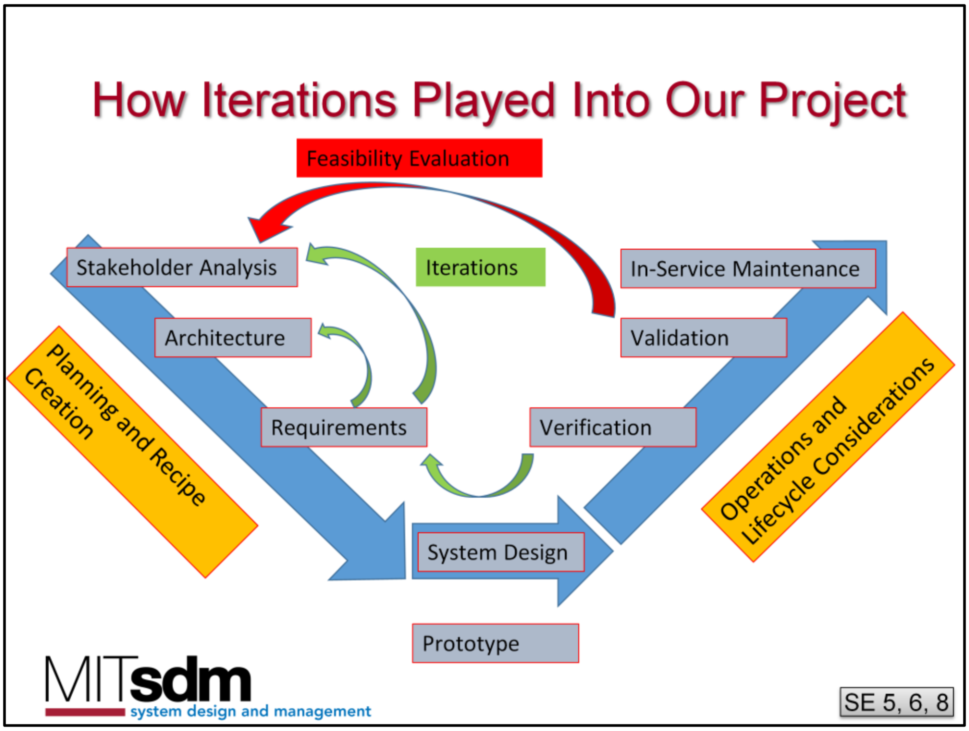
Level Indicator	(All)								
Count of Traceability Indicator	**Column Labels**								
Row Labels	**Appreciating**	**Creating_Plan**	**Delivering_Food**	**Distributing_Food**	**Obtaining_Funds**	**Preparing_Food**	**Sourcing_Food**	**(blank)**	**Grand Total**
A. Child		1		19				1	21
B. Apple-o-matic Enterprise		2							2
C. Sponsor	1	5			5			6	17
D. Donor	7				11			1	19
E. School		6	2	2		1	1	3	15
F. Community Volunteer	2				4			1	7
F. Sponsor	1								1
G. Apple-o-matic Social Franshise		1						2	3
H. Parent		3							3
J. Retailer							4	5	9
K. Teacher	1	1		5				4	11
L. Design Engineer		1							1
M. US Government								1	1
N. Apple Lobby				1					1
Grand Total	12	20	2	27	20	1	5	24	111

G. Apple-o-matic Social Franshise		1						2	3
H. Parent		3							3
Behaving Child		1							1
The System shall promote behavior benefits on website.		1							1
Healthy Child		1							1
The System shall promote health benefits on website.		1							1
Reduced Financial Pressure		1							1
The System shall promote financial benefits on website.		1							1
J. Retailer							4	5	9

Defining the high level requirements was the first step once we had established our goals and architecture. Questions like, what are the things that must be accomplished by each stakeholder, or rather, how is the system going to achieve the needs of each stakeholder in the system.

Once the high level stakeholder and requirements were identified, we were able to define lower level requirements. Each lower level requirement mapped to a higher level requirement, and correlated the requirement to the sequence in which it would impact.

SE1: System requirements including requirements traceability from stakeholder needs to subsystem and component requirements.



This describes how we have so far defined the planning, stakeholder analysis, and architecture methods used to develop the left side of the V. What we want to highlight here, is how at this point, we started to see the iterations that caused some of the final decisions we made. We saw that once we were forced to create the requirements matrix, we had to go back and review the stakeholders. We also had to review our planned architecture to make sure what we had written for requirements stood true for all stakeholders in the original plan.

We identified key subject matter experts to act as agents for our stakeholders.
We consulted with Juliana Cohen, Research Assocuiate at Harvard Department of Nutrition on healthy eating during school lunch programs.
We also consulted with Wendy Brannen with USApple.org to understand Apple producer needs.
In Dubuque Iowa we met with Carolyn Scherf with Iowa State University on local foods.
In Johnston Iowa we met with the school district. (More refinement needed.)

NEXT, we moved on to a verification plan, which included the feasibility study (shown soon). This forced us to think of the physical system, which opened up new questions not yet realized. This caused us to yet again, revisit this Requirements matrix and make final adjustments, which yet again made us think about the stakeholders, the priorities, and the architecture.

SE5: Verification and Validation Plan
SE6: Operations and lifecycle operations
SE8: Reflection on what your team have learned about systems thinking

Feasibility Evaluation Plan

Quality attributes to verify high risk architecture decisions:

- **Students hungry when provided apples**
- **Acceptance of the apples by students**
- **Responsive donor network for acquiring apples**

1. Load Basket with apples (22-25 small) | 2. Start App | 3. Distribute Apples

https://github.com/apple-o-matic/proto1/blob/master/system_engineering/Apple-o-matic_Feasibility_Evaluation.docx

MITsdm system design and management

PM 5

Aaron Speaking – (Need to trim to 1 minute) The Feasibility Evaluation consists of tests to verify high risk architecture decisions that overcome design challenges related to the function of distributing food (Step 6 in Concept of Operation). Using VDA Decision Process from Taming Diabetes *with Systems Thinking by* ***Carlos o. Morales***

The MIT SDM Team 25 conducted a field test using prototype software along with survey questions as part of the feasibility evaluation. The following important quality attributes were the areas of focus and desired outcomes are planned for this Feasibility Evaluation.

Students hungry when provided apples – Apple acceptance and impact of satiation depends on timeliness of feeding time. The desired outcome is to demonstrate hunger is a need by the child in the afternoon.

Acceptance of the apples by students – Apple rate of distribution is a challenge that may limit the viability of the solution. The desired outcome is for > 20% of students to take apples daily. (Based on DC3 intended outcome to increase student fruit consumption by 20% in participating school.)

Responsive donor network for acquiring apples – Apple rate of sourcing is a control input that varies based on identified recipients. The desired outcome is to demonstrate the donor network can be notified of student consumption changes with <1 day delay in the system to minimize the inventory. (Based on OS17 DSM analysis, we had already realized that a need must be identified prior to triggering donations, but not that recipients had to be identified prior to acquiring the food from the source in order to process for spoilage reasons.)

The Feasibility Evaluation shall consist of the following activities for each test day:
1) Daily survey administered by teachers to record student response
2) Use Apple-o-matic system proof-of-concept prototype to verify the feasibility around 2pm
Each student will perform the following steps to distribute the apples:
Open the basket lid
Hear a special educational message about apples
Take an apple to eat
Lower the lid for next student
3) Teacher Interview

Feasibility Evaluation Prototype

- Tablet App Created

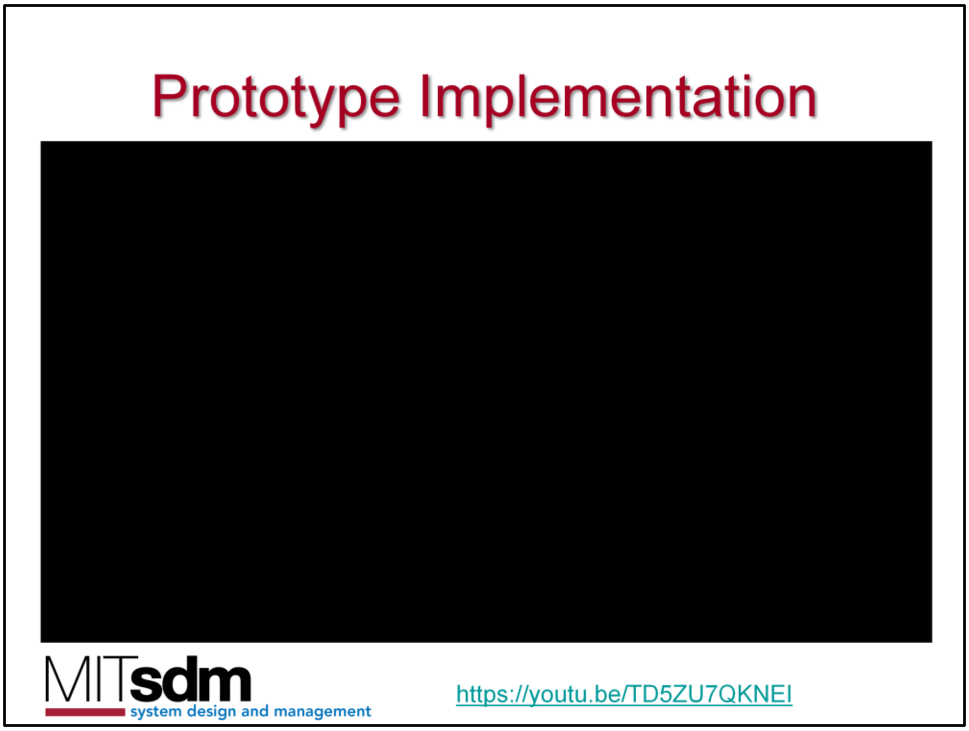
App Screenshots – Press “Go” to start “Apples Taken” counter for use in distribution with basket.

MITsdm system design and management

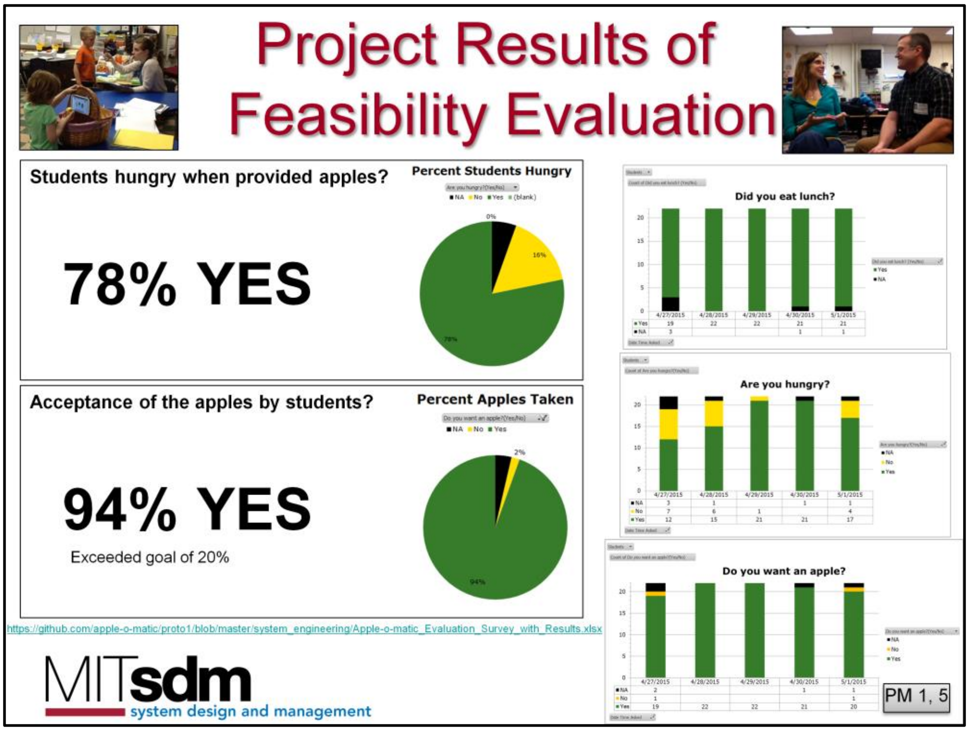
http://ai2.appinventor.mit.edu/#5775598984101888

Aaron Speaking – This app is an open source application created using the MIT app inventor tool for android devices.

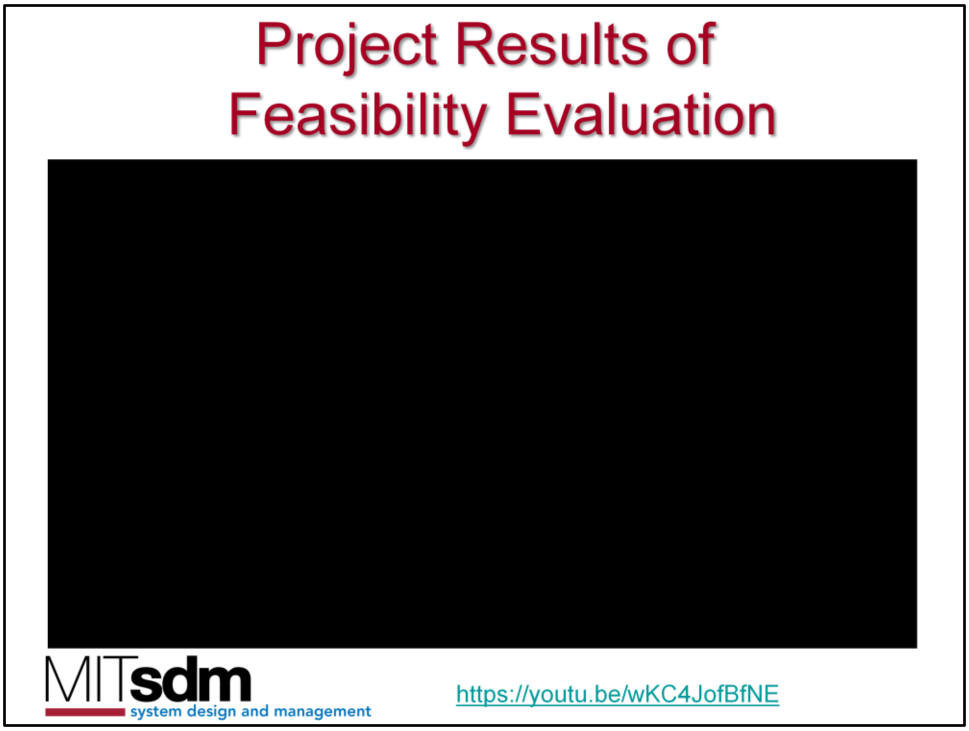
http://ai2.appinventor.mit.edu/#5775598984101888



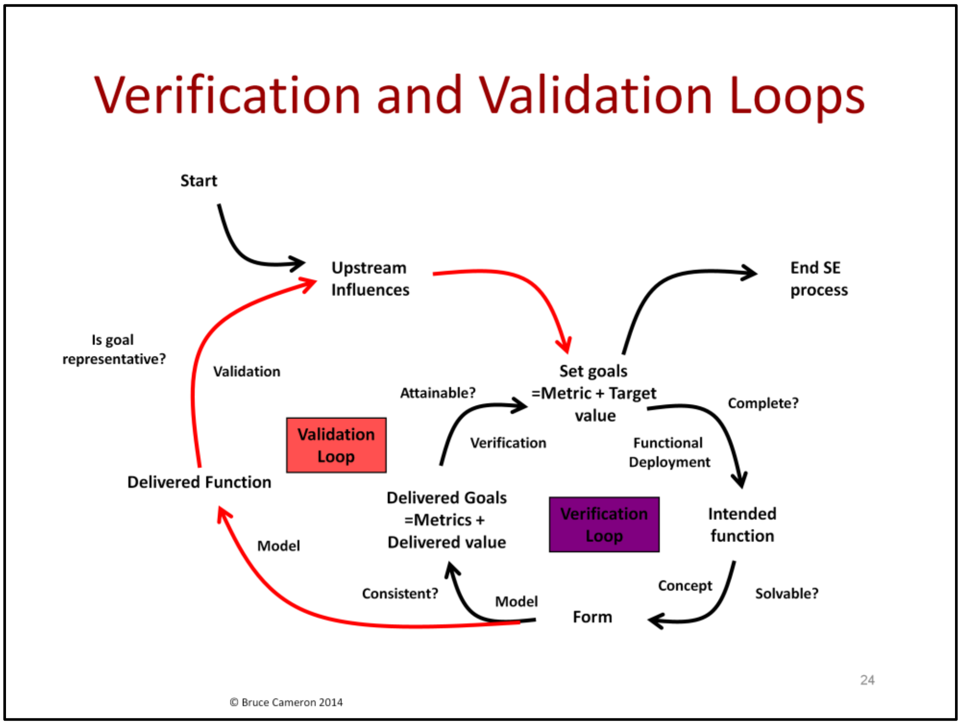
Aaron – Here’s an example of the a student taking an apple.



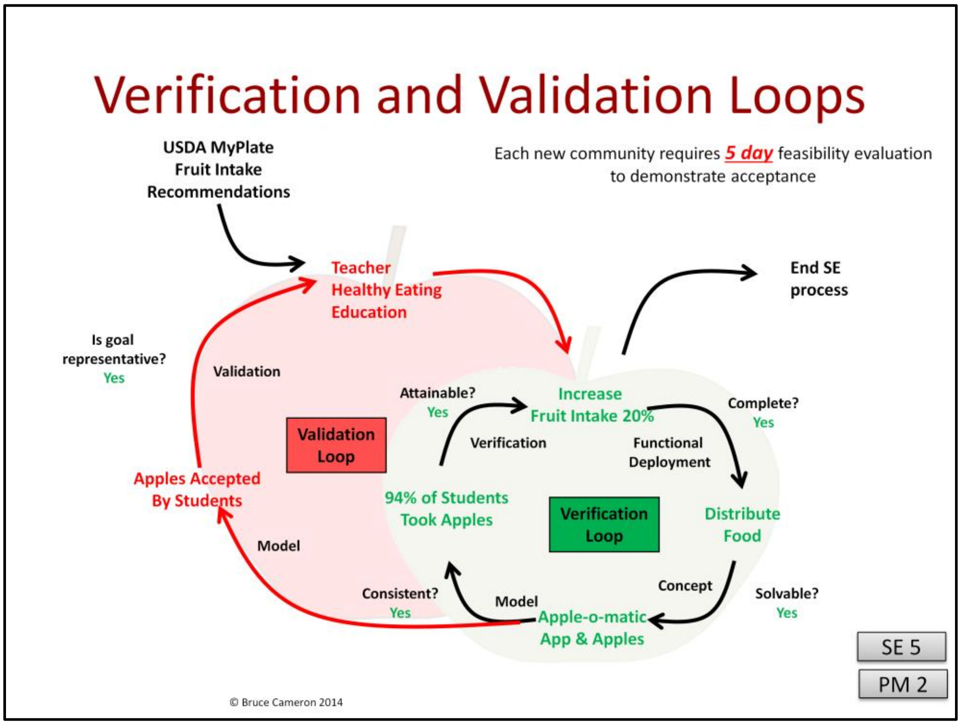
Aaron Speaking – The feasibility evaluation proved to be a big success. 78% of the students were hungry by 2pm even though they ate lunch around 11:30AM. We had a 94% of apples taken. Many apples were consumed immediately after to activity but some took them home to eat later. This successful experiment of a critical function with a proof-of-concept sets our project at Technology Readiness Level 3 (TRL 3) and *PM1: Targets (driven by hoped for value) and estimates (driven by analyses of feasibility) are shown and compared related to scope, project cost, and duration. PM5: Any design, prototyping, and validation tasks by the project team to date are reflected in an estimate of future implementation activities*



Aaron to introduce. – Original Video was 2 minutes and 43 seconds (Trimmed to 43 seconds)



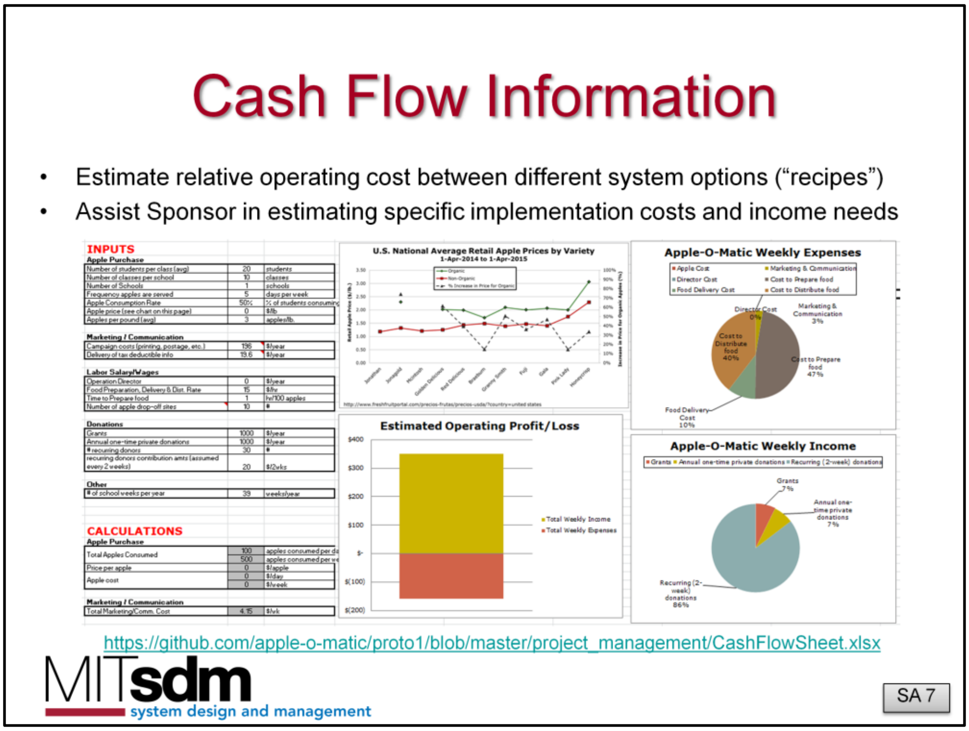
From our requirements, we defined a set of metrics we needed to achieve on a feasibility study to ensure we were on the right path. We had a goal of improving fruit intake for 20% of students using apple-o-matic. We used a feasibility evaluation to close the validation loop on this high risk architecture decision.



Aaron -

We had a goal of improving fruit intake for 20% of students using apple-o-matic. We used a feasibility evaluation to close the validation loop on this high risk architecture decision. This verification and validation loop should be repeated for each recipe solution created or modified for each release to system operations in a community.

Relate back to tree diagram as modular plan
SE5: Verification and Validation Plan
PM2: The expected scope of the project is clearly described, structured at a level of sustainable granularity, related to the product system and its value, and measureable.



Wes

Notes for speaking:

Continuing our move into the detailed implementation phase. . . .
Cash flow sheet – Project Deliverable
Aid in estimating relative system cost
Aid sponsor in estimating their costs and income required

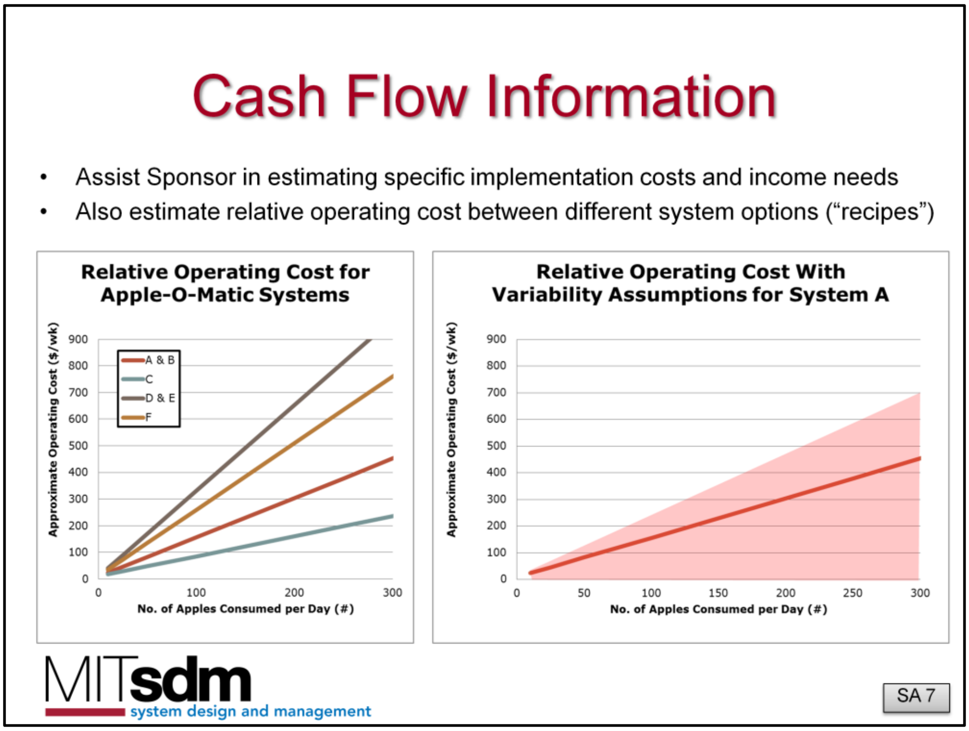
What it is:

This is a first-pass simulation of expected operation (cost focus) of the system.
This basic cash flow sheet provides a template, or “starting point” for a system implementer / community sponsor to begin analyzing income needs before committing to the project. This is a project deliverable from our team to the system implementer.
There is a separate setup (separate tab in Excel) with architectural assumptions baked in, but the implementer still needs to make some assumptions and statements about the implementation (generally non-architectural) to get the program going (volunteers vs paid staff, estimated apple consumption, etc.) and make program cost estimates.
The cash flow sheet considers areas of commodity (apple) purchases if necessary, Marketing and Communication, Labor (salary, wages, volunteers). Provides guidance on estimating certain inputs (for example, data provided for U.S. average retail apple prices by variety, etc.).

Why we made this:
This is a supporting piece of form to aid in system implementation. We wanted the implementer to be able to input their assumptions and "simulate" system cost responses. This was desired to ensure that the system implementer understands the scope of "what they're getting into", and can help understand system cost, and required donations/grants needed to support the system.
We also wanted a simple way to understand cost differentiation between the different systems. Originally, we made a qualitative analysis of the expected costs for the systems, but there was at least one instance where there were opposing costs that made a qualitative cost comparison between the architectures not clear (donated apples cost less to the system, but likely required more preparation in terms of washing, sorting, etc.). After making some simple cost assumptions, it was clear that the "preparation" activity cost much less than the cost of the actual apples.
Additionally, these cash flow sheets provided us (the design team) a simple operating cost comparison between the six different implementation "recipes" currently available. This was fed into the "Recipe Selection Guide" (a later slide in this presentation) as means to provide a high-level qualitative operating cost comparison on this flowchart ($ vs. $$$).

What this told us ("nuggets" of information):
When apples are purchased (not donated), they represent a large fraction of system operating cost.
The offsetting increased cost of handling donated apples is still minimal compared to the cost of having to buy apples (in other words, you would still rather get donated apples, when possible, even though it takes more effort to process them).
With only a few exceptions, apple retail prices don't vary much by variety, so buy the ones that the kids like best (or a varietal mix).
With the original assumptions, the defined recipes give an evenly spread cost variation.

SA7: An articulation of whether the architecture case and the business case close for this system, and if so how.



What it is:

This is a first-pass simulation of expected operation (cost focus) of the system. This basic cash flow sheet provides a template, or “starting point” for a system implementer / community sponsor to begin analyzing income needs before committing to the project. This is a project deliverable from our team to the system implementer.

There is a separate setup (separate tab in Excel) with architectural assumptions baked in, but the implementer still needs to make some assumptions and statements about the implementation (generally non-architectural) to get the program going (volunteers vs paid staff, estimated apple consumption, etc.) and make program cost estimates.

The cash flow sheet considers areas of commodity (apple) purchases if necessary, Marketing and Communication, Labor (salary, wages, volunteers). Provides guidance on estimating certain inputs (for example, data provided for U.S. average retail apple prices by variety, etc.).

Why we made this:

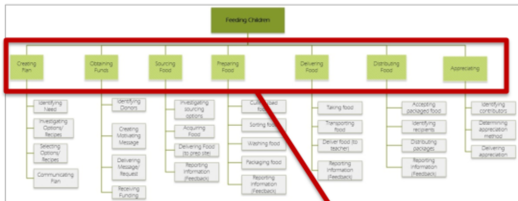
This is a supporting piece of form to aid in system implementation. We wanted the implementer to be able to input their assumptions and “simulate” system cost responses. This was desired to ensure that the system implementer understands the scope of “what they’re getting into”, and can help understand system cost, and required donations/grants needed to support the system.

We also wanted a simple way to understand cost differentiation between the different systems. Originally, we made a qualitative analysis of the expected costs for the systems, but there was at least one instance where there were opposing costs that made a qualitative cost comparison between the architectures not clear (donated apples cost less to the system, but likely required more preparation in terms of washing, sorting, etc.). After making some simple cost assumptions, it was clear that the "preparation" activity cost much less than the cost of the actual apples. Additionally, these cash flow sheets provided us (the design team) a simple operating cost comparison between the six different implementation "recipes" currently available. This was fed into the "Recipe Selection Guide" (a later slide in this presentation) as means to provide a high-level qualitative operating cost comparison on this flowchart ($ vs. $$$).

What this told us ("nuggets" of information):
When apples are purchased (not donated), they represent a large fraction of system operating cost.
The offsetting increased cost of handling donated apples is still minimal compared to the cost of having to buy apples (in other words, you would still rather get donated apples, when possible, even though it takes more effort to process them).
With only a few exceptions, apple retail prices don't vary much by variety, so buy the ones that the kids like best (or a varietal mix).
With the original assumptions, the defined recipes give an evenly spread cost variation.

SA7: An articulation of whether the architecture case and the business case close for this system, and if so how.

Architectural Decisions



- The differences in architectural solutions are a set of implementation variations (or "recipes") that are available to the project sponsor.

- Architectural impacts first identified by investigating system functional decomposition
- These impacts were then related to two architectural questions.

Impacted Area of Operations (Level 1 Decomposition)	Architectural Impact	Question: Source of apples (donated or purchased)?	Question: Which Distribution System Option (Simple, Complex, Manual, Automatic)
Creating Plan	None		
Obtaining Funds	Identifying Donors & Deliverying Message is a function of how much money is needed (function of what kind of system is selected).	X	X
Sourcing Food	Acquiring Food, Delivering Food, Reporting Information	X	
Preparing Food	Culling, Sorting, Washing, Packaging, Reporting	X	
Delivering Food	Who delivers food? To whom (multiple teachers, central food distribution, etc.)		X
Distributing Food	Identifying recipients, distributing packages, reporting information (Feedback)		X
Appreciating	Identifying Contributors (people deserving feedback depends on apple donations, monetary donations, both, etc.)	X	X

MITsdm system design and management

SA 1

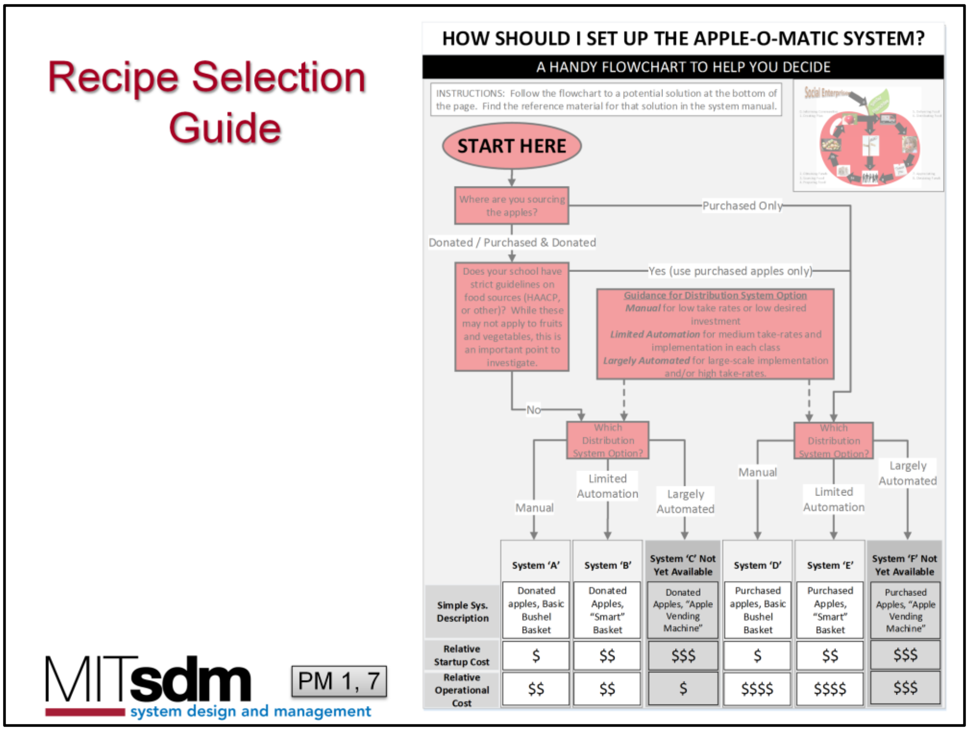
Notes for Speaking:
Not one architectural solution, but a set
First, investigate L1 decomp to identify sensitivities to expected implementation variations
Ask questions to help understand desired implementation variations
Map the two dimensions
This is the engineering view

Detailed Commentary:
In many Systems Architecture projects, the S.A. is required to make an architectural decision from a set of possibilities. In our case, we want to provide a ***set*** of architectural options ("recipes") to the community sponsor. In doing so, we want to enable the C.S. to make the appropriate architectural decision based on a simple interface with a minimal number of questions.

We began by analyzing our system (our functional decomposition and operational diagrams) and thinking through how the system may act or respond differently under different implementation scenarios. For each of the functional elements (at level 1 and/or 2), we asked questions about how those steps may vary. We wanted flexibility in the system, but simplicity in implementation as well (so as not to necessarily need a large number of "recipes" that need development and maintenance). Ultimately, the majority of architecturally impactful decisions can be determined by asking two questions listed in the table above “What is the source of your apples (purchased or donated)?”, and “Which kind of distribution system do you wish to use (simple, or automated)?”. In earlier implementations of this, there were 5 architectural questions, however the other three were determined to be less significant or ultimately important, but not architectural. Guidance for how to handle these other three cases is given in the formal instructions “recipe sheet” provided to the community sponsor.

With the two questions defined, we mapped those questions to the architecturally impactful areas previously defined, and demonstrated how the simple 2 questions supported making architectural decisions across the entire system (lower right-hand image above). With this mapping defined, we are now ready to give guidance to the community sponsor for their particular implementation (next slide).

SA1: A representation of the system architecture, including the architectural decisions that the team made and why.



Notes for Presentation:
This is the end-user version of the same sheet
Allows user to select a specific recipe/system, but also evaluate and compare

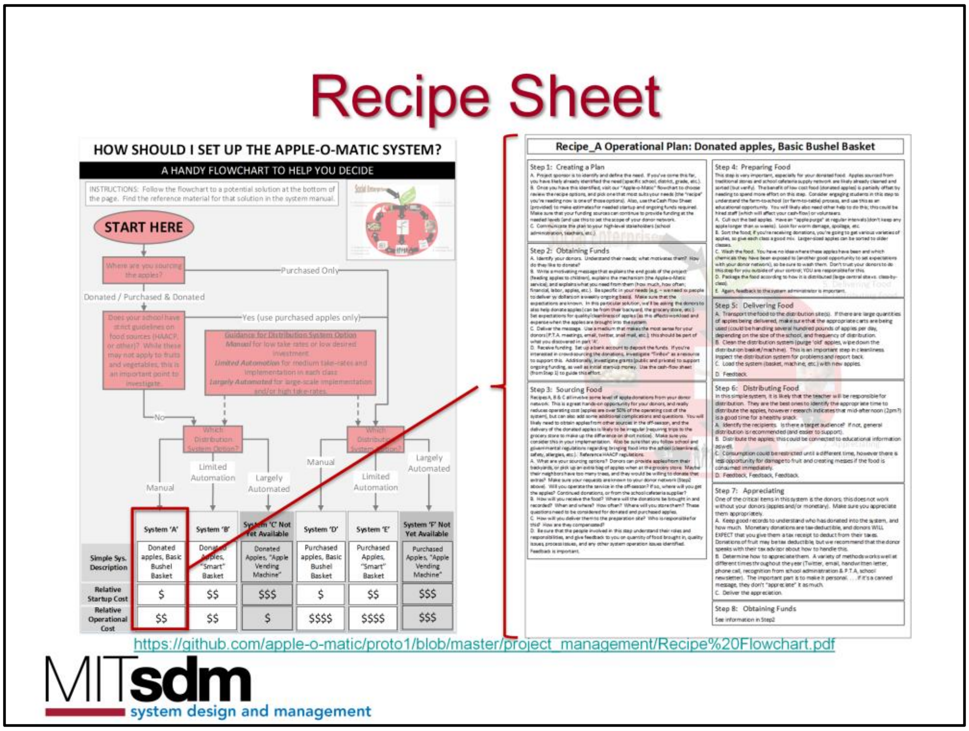
Detailed Commentary:
As mentioned in the previous slide, two architectural questions are asked that support the specific Sponsor's need to downselect and choose the appropriate architecture. These are outlined in a graphical form in the attached flowchart (The Apple-o-Matic system decision flowchart), which assists the implementer (community sponsor) in deciding which of the currently developed architectural options ("recipes") is best suited for their needs. This is also intended to be a project deliverable, given to the Sponsor as an implementation aid.

While there are only two architectural questions asked, we did add one additional "safety check" to this flowchart, as our SMEs indicated that certain regulations may limit the intake of "foreign" foods (outside of normal food acquisition network) into the schools for safety reasons. Because of this, we added an additional question to the flowchart, which highlights the need to investigate this important point.

Finally, a solution is selected for the Sponsor by answering the questions, however the benefit of this flowchart is that they can re-evaluate their solution/recipe against the other options clearly displayed and see if there might be a better option for their case. For example, estimated operating cost (from the cash flow sheet discussed earlier in this presentation) is compared between the four options, allowing the user to re-evaluate their needs and compare the different options.

The intent was to keep the number of recipes low to start with until other architecturally differentiating needs became more obvious (after a few months of system use), at which point, the previous slide (mapping of architectural differences to questions) would change, and this flowchart would necessarily change to accommodate those new needs.

PM1: Targets (driven by hoped for value) and estimates (driven by analyses of feasibility) are shown and compared related to scope, project cost, and duration.
PM7: Amongst a set of project scenarios, the team proposes a preferred and backup scenario for system implementation. If meaningful, these options are shown in a tradespace diagram. Concerns from stakeholders are anticipated and addressed.



Speaking Notes:

See Handout

Once architecture selected → specific recipe plan (see forms in front of you)

Maps to L1/2 decomp

Detailed Commentary:

Once the community sponsor has selected an appropriate system, they are guided to a particular detailed Operational Plan (another project deliverable). Shown here (on the right) is the operational plan for recipe ‘A’, which discusses Apple-o-Matic use in a situation where the apples are donated, and the distribution method is simple (basic bushel basket). The operational instructions are divided up into the seven boxes, each representing the level-1 functional elements. Each major section is put in a bulleted list that very closely reflects the level-2 functional decomposition as well. Guidance, recommendations and instructions are given at each step to the user. This one-page operational instruction sheet is a specific example of recipe ‘A’; additional operational plans would be available for each of the supported recipes.

Product Development Effort

Make your own Apps

Software Development			Sub-Totals	Duration (weeks)	Comments
Number of Software Engineers	2	engineers			1 Android OS developer/ 1 iOS developer
Number of System Engineers	1	engineer			5 years Requirements and Testing Experience
Hourly Rate	$50	$/hour			
Requirements Documentation using Agile	40	hours	$2,000	1	Define User Stories and Acceptance Criteria
Software Design	40	hours	$4,000	1	Software Engineers document screen mockups with flow
Software Coding	240	hours	$24,000	6	Software Engineers create code for smartphone and tablets
Software Automated Testing	160	hours	$16,000	4	Software Engineers create automated test for demos
Software Reviews/Demos	20	hours	$1,000	4	System Engineers Accept Completed User Stories
Development Environment	$5,000	$/engineer	$5,000		Mac/PC/Server/Software
System Testing					
Beta Release Field Testing	80	hours	$12,000	2	
Go to Production Field Testing	80	hours	$12,000	2	
Total Development Costs	$76,000	USD		5 months	

MITsdm system design and management

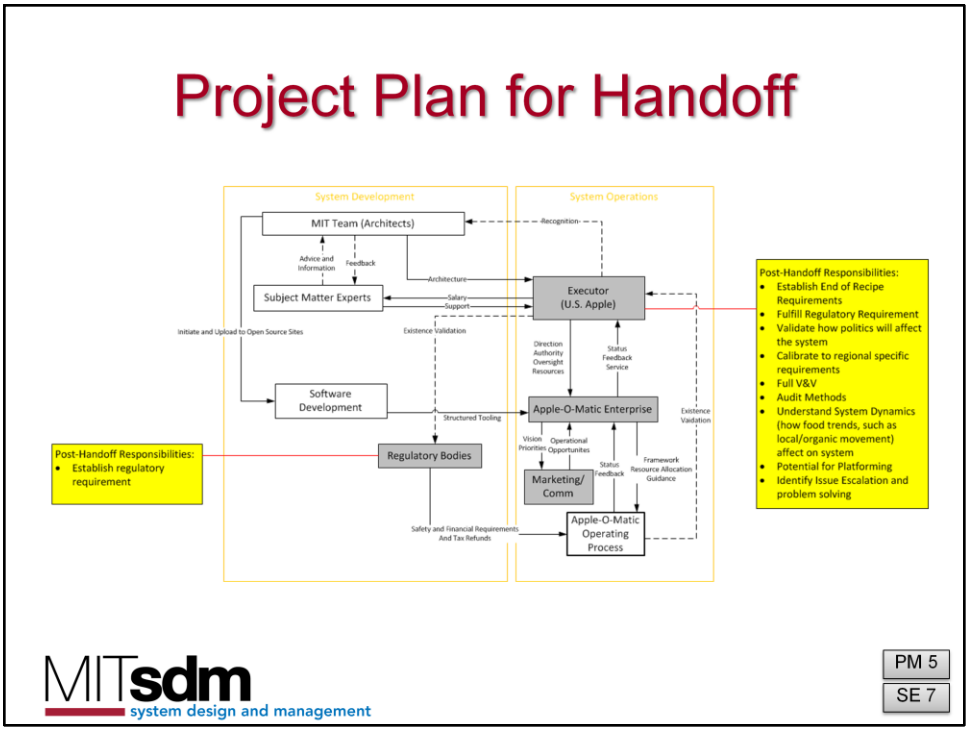
PM 1, 5

Wes

<u>Notes for Presentation:</u>

In addition to operational costs, also looked at development costs for new apps to support next recipe.

PM1: Targets (driven by hoped for value) and estimates (driven by analyses of feasibility) are shown and compared related to scope, project cost, and duration
PM5: Any design, prototyping, and validation tasks by the project team to date are reflected in an estimate of future implementation activities



Our team defined architecture and made the respective decisions to some level, however there are further decisions to be made

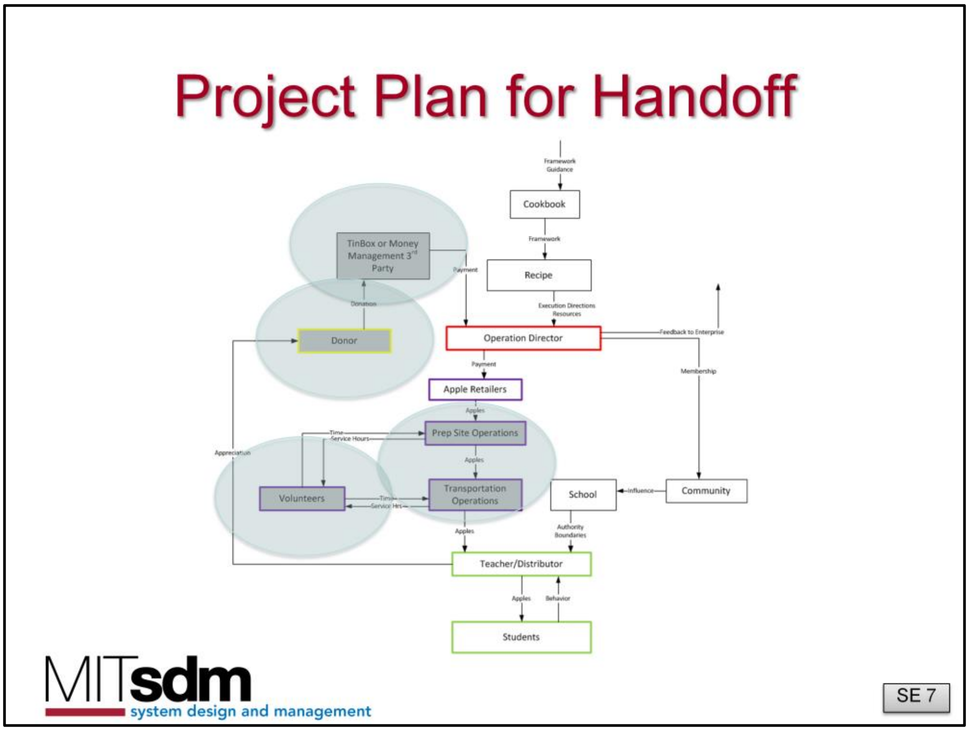
Final recommendations for the system to move forward in a higher level organized manner:

The Hierarchical System Control Structure serves to illustrate the broader context of influence on the operating process system.
To fully understand how the system would be controlled and operated from a decision making perspective and which elements influence others (global and local), this control loop diagram shows the views that we found to be of most importance.

Grey boxes indicate Customizable entities
Yellow is decisions to be made and further development to take place after entities are identified.

Within the larger system context, there (4) entities that have been identified as controllers, but are not rigidly defined (flexible/customizable). There is a level of dependence between the projected path of the project and the identification of these entities. For the project to proceed, the entities will have to perform items such as establish regulatory requirements (legal, standards, financial) depending on the organization which is hosting the project, establish what happens at the end of the actual recipe implementation, and after feedback loops are closed, model system dynamics over time (such as the impact of food trends) and establish other lifecycle implications. These items were either was not in scope of the project or are to be completed after identification of the indicated entities.

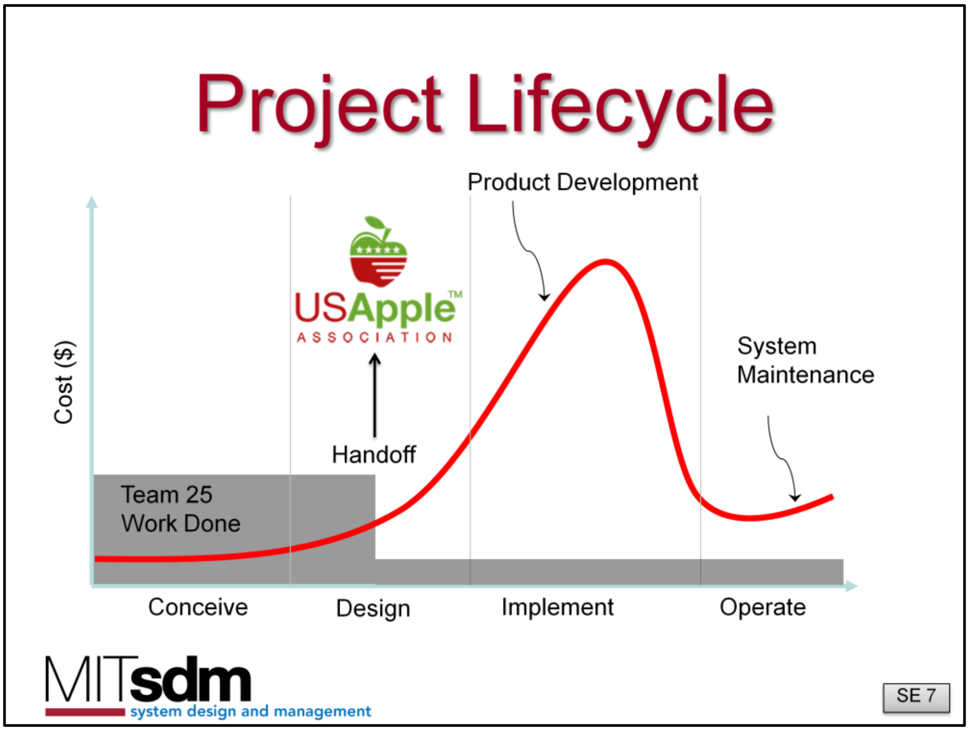
SE7: Final recommendations



Zooming in to the “operations process”, the grey circles indicate customizable entities that require further refinement at the implementation level. This further refinement is discussed in the detailed recipe sheet (so guidance on how to do this refinement is discussed). Example: How to acquire funds from donors (apps, bank transfer instructions, etc.), etc.

We also expect that over time as the “Enterprise” becomes more aware of specific implementation/customization needs, the list of recipes and corresponding instructions will be expanded.

SE7: Final recommendations



Wes:

Notes for presentation:

Development cost over time for new recipes (not operations)

Important points: Handoff to US Apple, feature development, operations

Recommendation that US Apple takes our simple recipe plan and implement it. Also evaluate software/app development cost against needs for this solution and willingness to support it at this time.

Detailed Commentary:

The intent of this diagram is to indicate project cost of the Apple-o-Matic Enterprise over time. This does not include operational costs for the specific instantiations of the individual operations, but rather higher-level “Enterprise” costs needed to develop new capabilities, new “recipes”, etc. This includes things like app development, automatic “vending machine” development, etc.

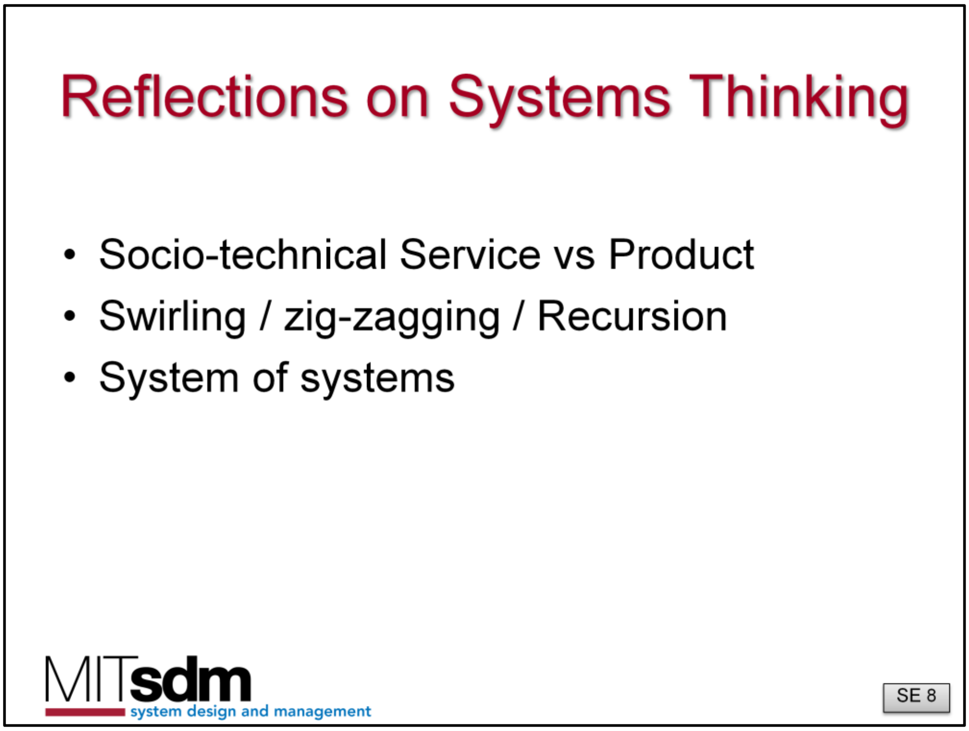
Note that our “concept” phase was not zero cost, that is because this cost includes our development time (priceless!), as well as some cost required to execute our PV&V activities (apple-o-matic test run in the Dubuque school system).

Conceive Design Implement Operate (CDIO) – based on http://www.cdio.org/

Our project utilized all stages of CDIO which we think will lower the overall PDP costs for USApple as we handoff.

Our recommendation is for USApple to bring the Apple-o-matic Service into production to increase apple sales and provide kids a nutritious snack.

SE7: Final recommendations



Wes

Notes for presentation:

We reflected back on what we learned and what we struggled with:

- Socio-technical systems (**Service**) are not far different from pure mechanical systems (**Product**)
- People are much more dynamic than mechanical systems
- Struggled with overlaid architecture, with swirling / zig-zagging (primary ben., architecture)
- Sys of Sys - The system ***depends*** on existing frameworks (vendors, transportation, etc.) ***Dependencies***

Acknowledgments

- John Helferich – Mentor
- Wendy Brannen – US Apple
- Juliana Cohen – Harvard Dept. of Nutrition
- Carolyn Scherf – Iowa State University
- Meredith Schmechel – Dubuque, IA Teacher
- Eric Toot – Johnston, IA Principal

MITsdm system design and management

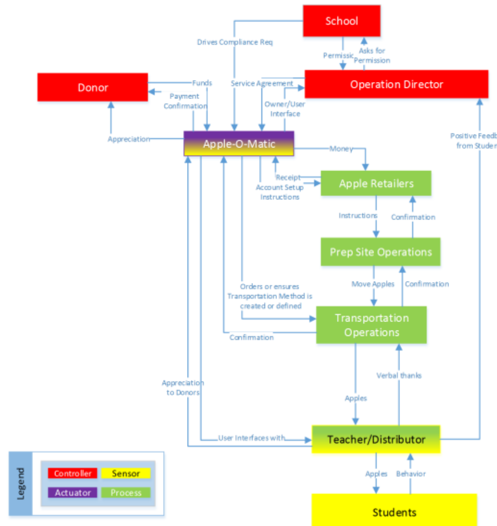
Backup Slides

- These slides are for reference and may be deleted prior to

MITsdm system design and management

Applying STPA

- STPA was used to find missing actions by groups within the system
- The accident "student not getting an apple" was our primary focus for this exercise
- Modified control loops to achieve SafetyHat logic and format

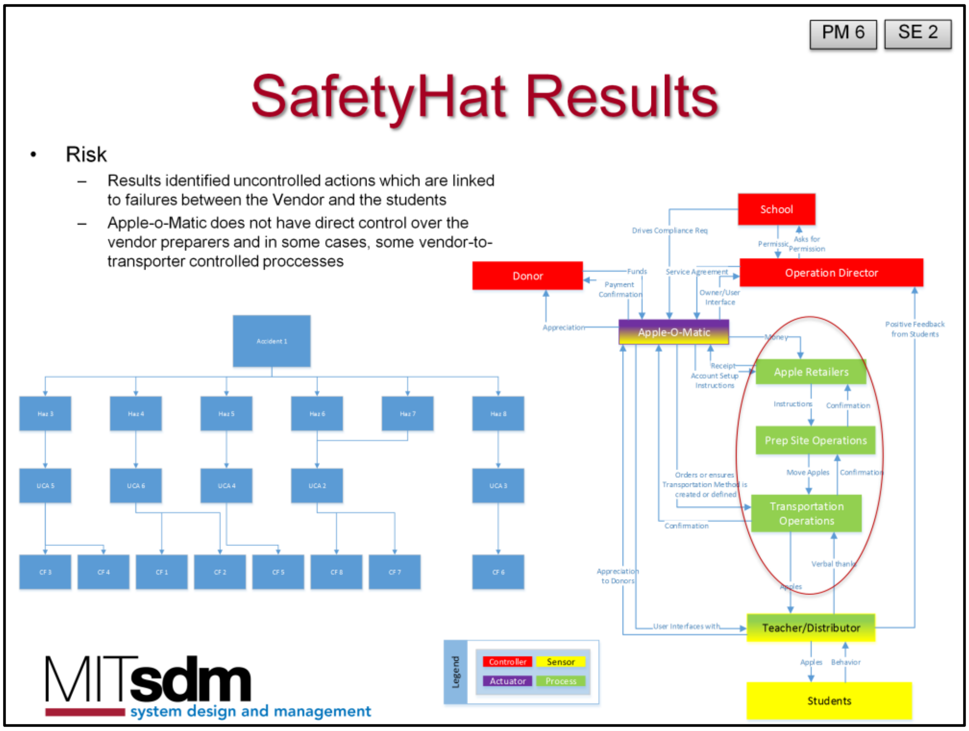


MITsdm
system design and management

PM 6 SE 2

SE2: Systems Engineering methods and tools used – why did you choose them and comments on methods you considered not appropriate for your project.

PM6: Risks and accepted mitigations are assessed and integrated in the project plan and estimates to completion. Uncertain work, re-work, and coordination activities are considered.



STPA is a bizarre tool for this analysis…. This is because STPA is used in the transportation industry to identify the Potential diffencies in a designed system that could result in an accident. We used STPA to try and identify hidden or missing control actions or linkages that could result in a system failure. The accident being that the students would NOT get a apple.

One thing that was interesting, was that the students forced the teachers to become a sensor after being the controlled process. Because the students do not directly interface with the apple-o-matic system, they are the primary beneficiary, and are the ones that create the highest value feedback. The teachers provide the sensor feedback on behalf of the students, through the actuator being the Apple-o-matic system. Which was another interesting point, because again we have a component in the control structure that can play two roles. Without separating or partitioning the apple-o-matic system into an actuator component AND a sensory component, the graphical representation above just holds two colors.

The hazards depicted (UAC 5, 2, 3, 6, 4, 2) are related to the controlled processes. The system does not take into consideration that there may be no apples to purchase (low season or peak purchasing), because right now we look at only one vendor per school per setup. If we could expand out to purchase apples from many vendors for the same day, you would avoid running into this issue.

Illustrated in the failure tree, the Hazards stemmed from one or a combination of controlled processes. The multiple or linked UCAs and CFs result from one or more linkages BETWEEN each of those controlled processes.

Also, transportation is very important. While it appears to be a benign process, timing is essential. If they miss the delivery window, the students may have already gone home, rending the days worth of apples useless (from a day to day perspective).

We chose to use this method because it helps identify weaknesses inherent in a design. We wanted to change the perspective in which we initially designed the apple-o-matic system, and find areas that would fail the system.